

dafan的专栏

目录视图

摘要视图

RSS 订阅

个人资料



dafan

访问： 46714次
积分： 727
等级：

BLOG 3

排名： 千里之外

原创： 30篇 转载： 8篇
译文： 0篇 评论： 8条

文章搜索

文章存档

2015年08月 (3)

2015年05月 (1)

2010年12月 (1)

2010年08月 (2)

2010年07月 (6)

展开

阅读排行

在 C/C++ 语言中特定的宏

(17209)

什么是RF、IF信号

(3935)

vc++_CTime 和 CString

(2399)

PCI Class Code

(2247)

How to be a good interviewee

(1513)

UltraEdit32 注册码

(858)

BCB6.0 父窗体最小化时

(754)

screen_capture keyboard events

(683)

101 Sample Job Interview Questions

(671)

如何在DOS中枚举PCI设备

(588)

评论排行

在 C/C++ 语言中特定的宏

(4)

Failed opening WinDrive

(1)

SendMessage参数详解

(1)

BCB6.0 父窗体最小化时

(1)

【CSDN会员专属福利】OpenStack Days China 大会门票，先到先得

【收藏】Html5 精品资源汇集

我们为什么选择Java

在 C/C++ 语言中特定的宏，如 __FUNCTION__

标签： [function](#) [语言](#) [编译器](#) [代码分析](#) [c](#) [date](#)

2010-08-01 21:49 17218人阅读 评论

版权声明： 本文为博主原创文章，未经博主允许不得转载。

__FILE__

__LINE__

__DATE__

__TIME__

__FUNC__

__FUNCTION__

在Visual Studio 2005中，默认情况下，此特性是激活的，但不能与/EP和/P编译选项同时使用。请注意在IDE环境中，不能识别__func__，而要用__FUNCTION__ 代替。

Comeau的用户也应使用 __FUNCTION__，而不是 __func__。

C++ BuilderX的用户则应使用稍稍不同的名字： __FUNC__。

GCC 3.0及更高的版本同时支持 __func__ 和__FUNCTION__。

+++++

仅仅为了获取函数名，就在函数体中嵌入硬编码的字符串，这种方法单调乏味还易导致错误，不如看一下怎样使用新的C99特性，在程序运行时获取函数名吧。

对象反射库、调试工具及代码分析器，经常会需要在运行时访问函数的名称，直到不久前，唯一能完成此项任务并且可移植的方法，是手工在函数体内嵌入一个带有该函数名的硬编码字符串，不必说，这种方法非常单调无奇，并且容易导致错误。本文将要演示怎样使用新的C99特性，在运行时获取函数名。

那么怎样以编程的方式从当前运行的函数中得到函数名呢？

答案是：使用__FUNCTION__ 及相关宏。

http://blog.csdn.net/dafan/article/details/5781491

1/4

- VC数字图像处理编程之一 (1)
- Windows多线程多任务设 (0)
- 个人成长的 15 种能力 (0)
- ShellExecute 妙用 (0)
- SQL遍历文件夹 / 文件, (0)
- keybd_event Src for win/ (0)

推荐文章

- *Android RoccoFix 热修复框架
- *RxJava 学习笔记（五） --- Creating 创建操作符
- *Android_GestureDetector手势滑动使用
- * 浅析ZeroMQ工作原理及其特点
- *Android开源框架Universal-Image-Loader基本介绍及使用
- *Spring Boot 实践折腾记（三）：三板斧，Spring Boot下使用Mybatis

最新评论

- 在 C/C++ 语言中特定的宏, 如 __DoDoMouse: 学习了
- 在 C/C++ 语言中特定的宏, 如 __boylinux: 赞!
- Failed opening WinDriver while t教兽: 请问你知道到哪能下到asset公司的scanworks软件吗? 谢谢.....
- 在 C/C++ 语言中特定的宏, 如 __peterli_xue:
- VC数字图像处理编程之一 dafan: This is a good kit for Digital picture, sharing wi...
- BCB6.0 父窗体最小化时子窗体才 dafan: 终于见识了一下,
- 在 C/C++ 语言中特定的宏, 如 __dafan:
- SendMessage参数详解 dafan: 以上, 来源于 http://msnvip.javaeye.com/blog/305 or ...

在 C/C++ 语言中特定的宏, 如 __FUNCTION__ - dafan的专栏 - 博客频道 - CSDN.NET

引出问题

通常, 在调试中最让人心烦的阶段, 是不断地检查是否已调用了特定的函数。对此问题的解决方法, 一般是添加一个cout或printf()——如果你使用C语言, 如下所示:

```
void myfunc()
{
    cout<<"myfunc()"<<endl;
    //其他代码
}
```

通常在一个典型的工程中, 会包含有数千个函数, 要在每个函数中都加入一条这样的输出语句, 无疑难过上“蜀山”啊, 因此, 需要有一种机制, 可以自动地完成这项操作。

获取函数名

作为一个C++程序员, 可能经常遇到 __TIME__、__FILE__、__DATE__ 这样的宏, 它们会在编译时, 分别转换为包含编译时间、处理的转换单元名称及当前时间的字符串。

在最新的ISO C标准中, 如大家所知的C99, 加入了另一个有用的、类似宏的表达式__func__, 其会报告未修饰过的 (也就是未裁剪过的)、正在被访问的函数名。请注意, __func__不是一个宏, 因为预处理器对此函数一无所知; 相反, 它是作为一个隐式声明的常量字符数组实现的:

```
static const char __func__[] = "function-name";
```

在function-name处, 为实际的函数名。为激活此特性, 某些编译器需要使用特定的编译标志, 请查看相应的编译器文档, 以获取具体的资料。

有了它, 我们可免去大多数通过手工修改, 来显示函数名的苦差事, 以上的例子可如下所示进行重写:

```
void myfunc()
{
    cout<<"__FUNCTION__"<<endl;
}
```

官方C99标准为此目的定义的__func__标识符, 确实值得大家关注, 然而, ISO C++却不完全支持所有的C99扩展, 因此, 大多数的编译器提供商都使用 __FUNCTION__ 取而代之, 而 __FUNCTION__ 通常是一个定义为 __func__ 的宏, 之所以使用这个名字, 是因为它已受到了大多数的广泛支持。

在Visual Studio 2005中, 默认情况下, 此特性是激活的, 但不能与/EP和/P编译选项同时使用。请注意在IDE环境中, 不能识别__func__, 而要用__FUNCTION__ 代替。

Comeau的用户也应使用 __FUNCTION__, 而不是 __func__ 。

C++ BuilderX的用户则应使用稍稍不同的名字: __FUNC__ 。

GCC 3.0及更高的版本同时支持 __func__ 和__FUNCTION__ 。

一旦可自动获取当前函数名, 你可以定义一个如下所示显示任何函数名的函数:

```
void show_name(const char * name)
{
    cout<<name<<endl;
```

```
}

void myfunc()
{
    show_name(__FUNCTION__); //输出: myfunc
}

void foo()
{
    show_name(__FUNCTION__); //输出: foo
}
```

因为 __FUNCTION__ 会在函数大括号开始之后就立即初始化，所以，foo()及myfunc()函数可在参数列表中安全地使用它，而不用担心重载。

签名与修饰名

__FUNCTION__ 特性最初是为C语言设计的，然而，C++程序员也会经常需要有关他们函数的额外信息，在 Visual Studio 2005中，还支持另外两种非标准的扩展特性：__FUNCDNAME__ 与 __FUNCSIG__，其分别转译为一个函数的修饰名与签名。函数的修饰名非常有用，例如，在你想要检查两个编译器是否共享同样的ABI时，就可派得上用场，另外，它还能帮助你破解那些含义模糊的链接错误，甚至还可用它从一个DLL中调用另一个用C++链接的函数。在下例中，show_name()报告了函数的修饰名：

```
void myfunc()
{
    show_name(__FUNCDNAME__); //输出: ?myfunc@@YAXXZ
}
```

一个函数的签名由函数名、参数列表、返回类型、内含的命名空间组成。如果它是一个成员函数，它的类名和const/volatile限定符也将是签名的一部分。以下的代码演示了一个独立的函数与一个const成员函数签名间的不同之处，两个函数的名称、返回类型、参数完全相同：

```
void myfunc()
{
    show_name(__FUNCSIG__); // void __cdecl myfunc(void)
}

struct S
{
    void myfunc() const
    {
        show_name(__FUNCSIG__); //void __thiscall S::myfunc(void) const
    }
};
```

顶 踩

1

0

上一篇 BCB6.0 父窗体最小化时子窗体关闭不了

下一篇 vc++_CTime 和 CString 之间的转换

猜你在找

- iOS8-Swift开发教程

C++语言基础

老郭全套iOS开发课程【Objective-C】

《C语言/C++学习指南》语法篇（从入门到精通）

微信公众平台开发入门
- 嵌入式C/C++语言精华文章集锦

c程序编译

Member Function Pointers and the Fastest Possible

c语言的编写格式

C错误大全

极光推送

消息推送领导品牌全面升级

JIGUANG | 极光

查看评论

4楼 DoDoMouse 2013-12-04 22:36发表




学习了

3楼 boylinux 2013-09-13 14:24发表




赞！

2楼 peterli_xue 2011-06-12 23:25发表



[e01]

1楼 dafan 2010-08-01 22:27发表



[e01][e10]

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

- 全部主题

VPN

BI

Splashtop

coremail

Compuware

Angular
- Hadoop

Spark

HTML5

UML

OPhone

大数据

Cloud Foundry
- AWS

IE10

Apache

components

CouchBase

aptech

Redis
- 移动游戏

Eclipse

.NET

Windows Mobile

云计算

Perl

Scala
- Java

CRM

API

HTML

iOS6

Tornado

Django
- Android

JavaScript

SDK

QEMU

Rackspace

Ruby

Bootstrap
- iOS

数据库

IIS

KDE

Web App

Hibernate
- Swift

智能硬件

Fedora

Cassandra

ThinkPHP
- Docker

NFC

XML

LBS

HBase
- OpenStack

jQuery

Unity

CloudStack

Pure
- FTC

Maemo

Solr

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 |

江苏乐知网络技术有限公司 提供商务支持

京 ICP 证 09002463 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved

